Task 3:- Using clauses, operations & Functions in queries processing on database for different DML, DRL operations revival results of quries using indent functions using ~~using~~ aggregatee, date, string, set clauses & operators. answers for this using employee database.

Employee databox

| Emp-id | Emp-name | Dept | Salary | Joining date | City. |
|---|---|---|---|---|---|
| 101 | Karthik | IT | 70000.00 | 2023-05-01 | ~~Madras~~ Chennai |
| 102 | Vinay kumar | HR | 55000.00 | 2018-03-15 | Hyderabad |
| 103 | Modit | Finance | 80000.00 | 2016-11-23 | Delhi |
| 104 | Nazeem | IT | 75000.00 | 2020-07-10 | Vizag |
| 105 | Bhanuprasad | Marketing | 60000.00 | 2019-02-28 | Vijayawada |
| 106 | Chaitanya | IT | 72000.00 | 2021-08-16 | Bangalore |
| 107 | Rohit sai | HR | 52000.00 | 2017-04-05 | Chennai |
| 108 | Akhil | Finance | 81000.00 | 2015-09-30 | Vizag. |
| 109 | Kailash | Marketing | 63000.00 | 2022-01-12 | Mumbai |
| 110 | Harsha | IT | 68000.00 | 2024-06-19 | Chennai |

3.1 Perform DML operands.

a) Insert new employee.

```
INSERTINTO Employee ~~(Employee id,~~
  (EMPId, EmpName ,Dept , salary, Joining Date, city)
VALUES (111,'sophiaa Green', 'Finance','72000.00',
  '2024-03-10', 'Houston');
```

b) Update Salary of employee in It department by 10%

```
UPDATE Employee
SET Salary = salary * 1.10
WHERE Dept = 'IT';
```

c) Delete employees who joined before 2015

```
DELETE FROM Employee.
WHERE Joining Data < '2015-01-01';
```

3.2 DRL queries using clauses, operator, & Functions

a) Retrieve employees with salary above average salary.

```
SELECT Employe-name, salary.
FROM Employee.
WHERE Salary > (SELECT AVG(salary) FROM Employee);
```

b) Retrive total salary per department.

```
SELECT Dept, sum (salary) As Total salary
FROM Employee
GROUP BY Dept;
```

c) Display employees with their years of service.

```
SELECT EmpName, TIMESTAMDIFF(YEAR, Joining
       Date, CUR DATE())
AS YEAR of service
FROM Employee.
```

d) Retrive employee whose name starts with 'A'.

```
SELECT *
FROM Employee
WHERE EmpName LIKE 'A%'.
```

e) Retrive employees joined in the last 2 years.

```
SELECT *
FROM EMPLOYEE
WHERE Joining Date >= DATE_SUB (CUR DATE()
      INTERVAL 2YEARS);
```

f) Use CASE Operator to classify employees by Salary.

```
SELECT Emplayee name, saleray
  CASE
     WHEN Salary> =8000 THEN 'High Salary'
     WHEN Salary Between 6000 AND 79999 THEN
     'Medium Salary'
      ELSE 'low salary'.
   END AS SALARY Category
 FROM Employee;
```

3.3 Set operators Examples (using two tables: Employee & New Employee).

a) Combine employees both tables without duplicates (UNION)

```
SELECT Employee FROM New Employee;
UNION
SELECT Employee FROM New Employee;
```

b) Find employee common in both tables (Interset)

```
SELECT Employeename.
FROM Employee
WHERE Employee IN(SELECT Employee FROM
   New Employee);
```

c) Find employees in Employee but not in New employee (MINUS/EXCEPT)

```
SELECT Empname
 FROM Employee
 WHERE Empname NOT IN(SELECT Name FR
   New employee);
```

3.4 Using string Functions:

a) Concatenate employee name & city.

```
SELECT Concate (Empnaime,' ', city) As Name with city.
FROM Employee;
```

b) Find employee with name length grather than 6.

```
SELECT Empname
FROM Employee.
WHERE Length (Employee name) > 6;
```

| VELTECH | |
|---|---|
| PERFORMANCE (5) | 5 |
| RESULT AND ANALYSIS (3) | 3 |
| VIVA VOCE (3) | 3 |
| RECORD (4) | 4 |
| TOTAL (15) | 15 |
| SIGN WITH DATE | [illegible] 13/10/23 |

Result:- Using clases, operation & Function in query is proved.

[illegible] 13/10/23